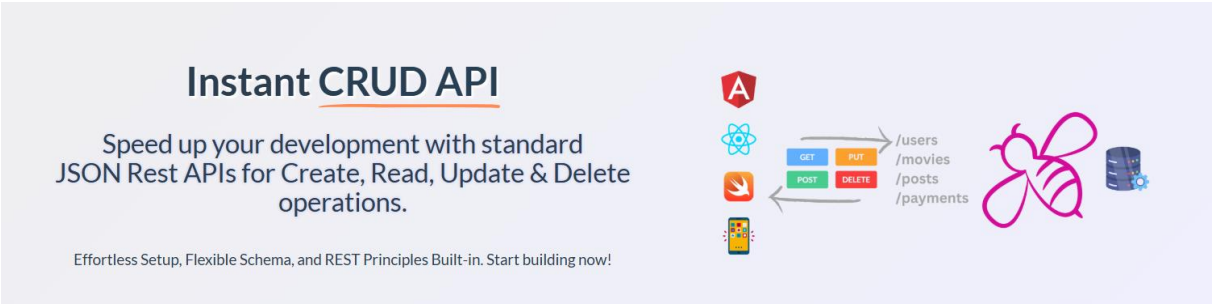
Exercice : Gestion des Utilisateurs avec API REST

https://kbma.github.io/usersManagerJS/users.html

Objectif :

Cet exercice vous permettra de mettre en place une interface web pour gérer une liste d'utilisateurs en utilisant l'API REST. Vous devrez créer un formulaire permettant d'ajouter, modifier et supprimer des utilisateurs. Vous utiliserez JavaScript pour interagir avec l'API et manipuler les données sur la page HTML.

Pour créer API CRUD temporellement : https://beeceptor.com/crud-api/



Énoncé de l'exercice :

Vous devez créer une page web qui permet de gérer une liste d'utilisateurs en utilisant une API REST. Cette page doit permettre les actions suivantes :

1. **Ajouter un utilisateur** : En remplissant un formulaire avec le nom et l'email de l'utilisateur et en cliquant sur le bouton "Ajouter".
2. **Modifier un utilisateur** : En affichant un formulaire pré-rempli avec les informations de l'utilisateur sélectionné et en cliquant sur le bouton "Mettre à jour".
3. **Supprimer un utilisateur** : En cliquant sur le bouton "Supprimer" à côté de chaque utilisateur de la liste.
4. **Afficher la liste des utilisateurs** : Au premier chargement de la page, la liste des utilisateurs sera récupérée via une API REST et affichée dans un tableau HTML.

API fournie :

Voici les points de l'API REST que vous utiliserez dans cet exercice :

Méthode HTTP	Path de l'API
GET	https://ca8d94623e86345abdd4.free.beeceptor.com/api/users
GET	https://ca8d94623e86345abdd4.free.beeceptor.com/api/users/:id
POST	https://ca8d94623e86345abdd4.free.beeceptor.com/api/users
PUT	https://ca8d94623e86345abdd4.free.beeceptor.com/api/users/:id
DELETE	https://ca8d94623e86345abdd4.free.beeceptor.com/api/users/:id

Instructions :

1. **Création du formulaire** :
   - Le formulaire devra contenir des champs pour le nom et l'email de l'utilisateur. Vous ajouterez deux boutons : un pour ajouter un utilisateur, et un autre pour mettre à jour les informations d'un utilisateur sélectionné.
2. **Affichage des utilisateurs** :
   - Vous devez créer un tableau HTML pour afficher la liste des utilisateurs avec leurs informations (nom et email). Chaque ligne du tableau doit comporter un bouton "Modifier" pour pré-remplir le formulaire avec les données de l'utilisateur sélectionné, et un bouton "Supprimer" pour supprimer cet utilisateur.
3. **Ajouter un utilisateur** :
   - Lorsque l'utilisateur clique sur "Ajouter", les données du formulaire seront envoyées à l'API via une requête `POST`. Après l'ajout, la liste des utilisateurs doit être mise à jour automatiquement.
4. **Modifier un utilisateur** :
   - Lorsque l'utilisateur clique sur "Modifier", les informations de l'utilisateur sélectionné doivent être affichées dans le formulaire. Après avoir modifié les informations et cliqué sur "Mettre à jour", les données doivent être envoyées à l'API via une requête `PUT`.
5. **Supprimer un utilisateur** :
   - Lorsqu'un utilisateur clique sur "Supprimer", un message de confirmation doit apparaître. Si la confirmation est positive, la requête `DELETE` sera envoyée pour supprimer l'utilisateur.

Étape 1 : Structure de la page HTML

Énoncé : La première étape consiste à préparer la structure de base de la page avec HTML et à inclure les dépendances nécessaires.

Code :

```
<!DOCTYPE html>
<html lang="en">
```

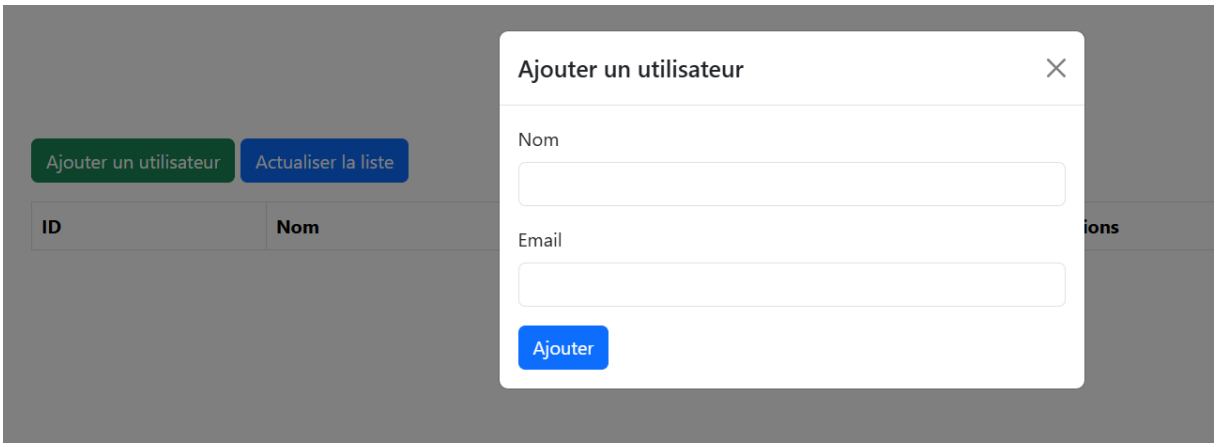
```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Gestion des Utilisateurs</title>
  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet">
</head>
<body>
  <div class="container mt-5">
    <h1 class="text-center mb-4">Gestion des Utilisateurs</h1>
    <!-- Boutons pour ajouter et actualiser -->
    <button class="btn btn-success mb-3" data-bs-toggle="modal" data-bs-target="#addUserModal">Ajouter un utilisateur</button>
    <button class="btn btn-primary mb-3" onclick="fetchUsers()">Actualiser la liste</button>
    <!-- Table des utilisateurs -->
    <table class="table table-bordered">
      <thead>
      <tr>
        <th>ID</th>
        <th>Nom</th>
        <th>Email</th>
        <th>Actions</th>
      </tr>
      </thead>
      <tbody id="userTableBody">
        <!-- Les utilisateurs seront chargés ici dynamiquement -->
      </tbody>
    </table>
  </div>
  <!-- Bootstrap JS -->
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

Explication :

- Cette structure de base comprend le titre de la page et un conteneur principal avec deux boutons : un pour ajouter un utilisateur et l'autre pour actualiser la liste des utilisateurs.
- La table vide sert à afficher dynamiquement la liste des utilisateurs une fois qu'ils sont chargés depuis l'API.
- Le fichier Bootstrap est inclus pour styliser la page.

Étape 2 : Création des modaux pour ajouter, modifier et supprimer un utilisateur

Énoncé : Nous allons ajouter les fenêtres modales pour permettre l'ajout, la modification et la suppression d'utilisateurs.



Code :

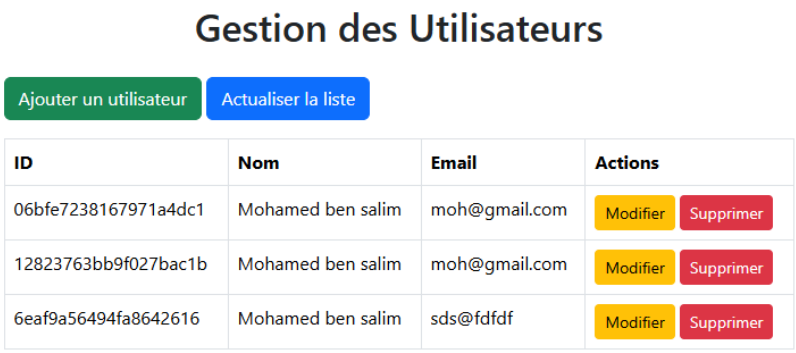
```
<!-- Modal Ajouter Utilisateur -->
<div class="modal fade" id="addUserModal" tabindex="-1" aria-labelledby="addUserModalLabel" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="addUserModalLabel">Ajouter un utilisateur</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
      </div>
      <div class="modal-body">
        <form id="addUserForm">
          <div class="mb-3">
            <label for="addUserName" class="form-label">Nom</label>
            <input type="text" class="form-control" id="addUserName" required>
          </div>
          <div class="mb-3">
            <label for="addUserEmail" class="form-label">Email</label>
            <input type="email" class="form-control" id="addUserEmail" required>
          </div>
          <button type="submit" class="btn btn-primary">Ajouter</button>
        </form>
      </div>
    </div>
  </div>
</div>
```

Explication :

- Un modal est ajouté pour permettre à l'utilisateur de saisir un nom et un email pour ajouter un utilisateur.
- Le formulaire contient des champs pour le nom et l'email, et un bouton pour envoyer les données.

Étape 3 : Interaction avec l'API pour charger les utilisateurs

Énoncé : Nous allons maintenant écrire la fonction JavaScript pour récupérer les utilisateurs depuis une API et les afficher dans la table.

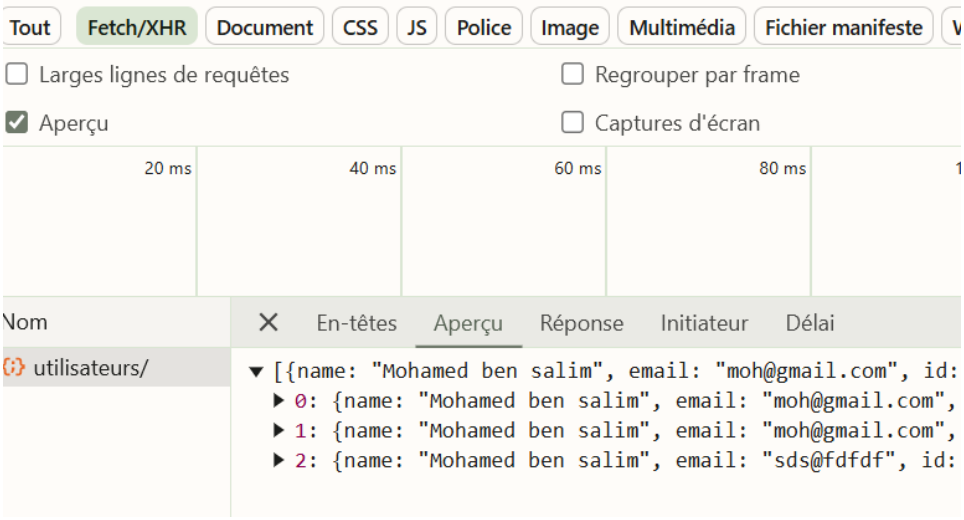


Code :

```
 const apiBaseUrl = "https://ca33fcdf4f239376367d.free.beeceptor.com/api/utilisateurs";
  const userTableBody = document.getElementById("userTableBody");

function escapeString(str) {
    return str.replace(/'/g, "\\'").replace(/"/g, '\\"');
  }


async function fetchUsers() {
  try {
    const response = await fetch(apiBaseUrl);
    const users = await response.json();
    renderUsers(users);
  } catch (error) {
    console.error("Erreur lors du chargement des utilisateurs :", error);
  }
}

function renderUsers(users) {
  userTableBody.innerHTML = users.map(user => `
    <tr>
      <td>${user.id}</td>
      <td>${user.name}</td>
      <td>${user.email}</td>
      <td>
        <button class="btn btn-warning btn-sm" onclick="showEditModal('${user.id}',
'${escapeString(user.name)}', '${escapeString(user.email)}')">Modifier</button>
        <button class="btn btn-danger btn-sm" onclick="showDeleteModal('${user.id}')">Supprimer</button>
      </td>
    </tr>
  `).join('');
}
```

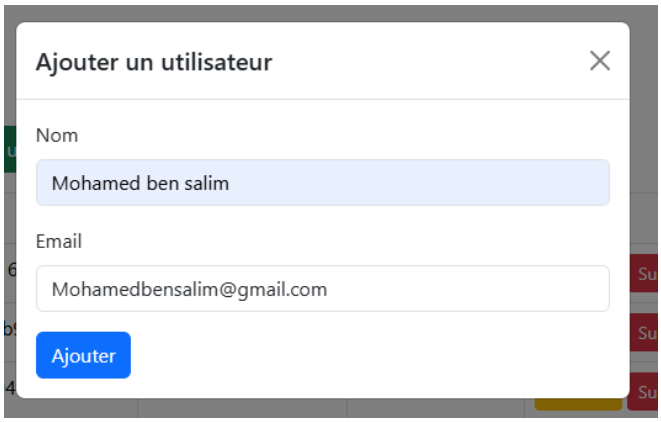
Explication :

- La fonction `fetchUsers` récupère les utilisateurs depuis l'API et appelle la fonction `renderUsers` pour afficher les données dans la table.
- La fonction `renderUsers` utilise la méthode `map()` pour créer une ligne pour chaque utilisateur et insère cette ligne dans le `tbody` de la table.



Étape 4 : Ajouter un utilisateur

Énoncé : Nous devons maintenant permettre l'ajout d'un utilisateur via un formulaire.



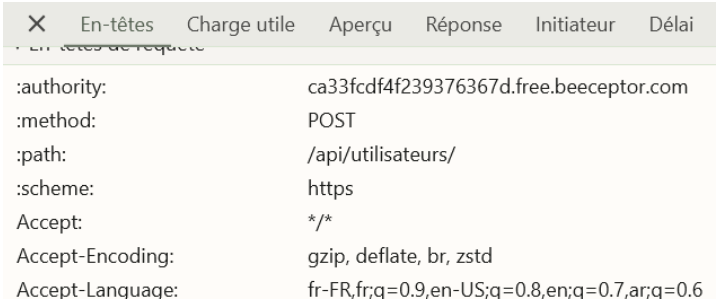
Code :

```
  const addUserForm = document.getElementById("addUserForm");
  const addModal = new bootstrap.Modal(document.getElementById("addUserModal"));

addUserForm.onsubmit = async function (e) {
  e.preventDefault();
  const name = document.getElementById("addUserName").value;
  const email = document.getElementById("addUserEmail").value;

  try {
    await fetch(apiBaseUrl, {
      method: "POST",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify({ name, email })
    });
    addModal.hide();
    fetchUsers();
  } catch (error) {
    console.error("Erreur lors de l'ajout d'un utilisateur :", error);
  }
```

```
};
```

Explication :

- Lorsque le formulaire d'ajout est soumis, nous récupérons les données saisies et les envoyons à l'API avec une requête POST.
- Après l'ajout de l'utilisateur, nous actualisons la liste des utilisateurs en appelant `fetchUsers()`.



Étape 5 : Modifier un utilisateur

Énoncé : Nous allons maintenant permettre de modifier un utilisateur existant.

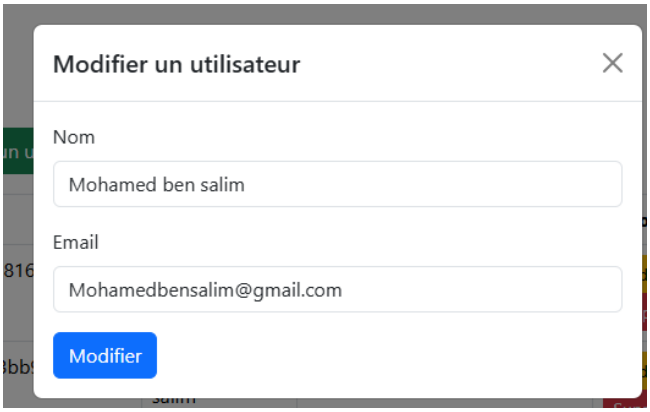
1/ Ajouter le model du formulaire editUserModal

```
<!-- Modal Modifier Utilisateur -->
<div class="modal fade" id="editUserModal" tabindex="-1" aria-labelledby="editUserModalLabel" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="editUserModalLabel">Modifier un utilisateur</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
      </div>
      <div class="modal-body">
        <form id="editUserForm">
          <input type="hidden" id="editUserId">
          <div class="mb-3">
            <label for="editUserName" class="form-label">Nom</label>
            <input type="text" class="form-control" id="editUserName" required>
          </div>
          <div class="mb-3">
            <label for="editUserEmail" class="form-label">Email</label>
            <input type="email" class="form-control" id="editUserEmail" required>
          </div>
```

```
            <button type="submit" class="btn btn-primary">Modifier</button>
          </form>
        </div>
      </div>
    </div>
</div>
```

2/Afficher le modal de modification

```
const editModal = new bootstrap.Modal(document.getElementById("editUserModal"));

    function showEditModal(id, name, email) {
      document.getElementById("editUserId").value = id;
      document.getElementById("editUserName").value = name;
      document.getElementById("editUserEmail").value = email;
      editModal.show();
    }
```

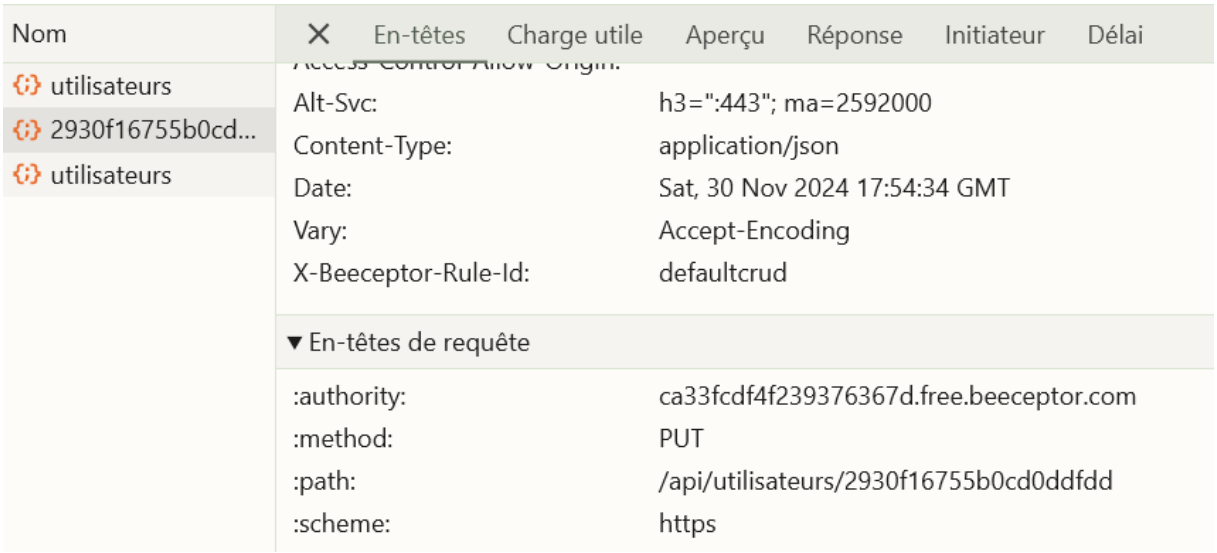


3/Code de l'action de la modification :

```
editUserForm.onsubmit = async function (e) {
  e.preventDefault();
  const id = document.getElementById("editUserId").value;
  const name = document.getElementById("editUserName").value;
  const email = document.getElementById("editUserEmail").value;

  try {
    await fetch(`${apiBaseUrl}/${id}`, {
      method: "PUT",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify({ name, email })
    });
    editModal.hide();
    fetchUsers();
  } catch (error) {
    console.error("Erreur lors de la modification d'un utilisateur :", error);
  }
};
```

Explication :

- Nous récupérons l'ID de l'utilisateur à modifier et envoyons une requête PUT à l'API pour mettre à jour les informations de l'utilisateur.
- La liste des utilisateurs est mise à jour après la modification.



Étape 6 : Supprimer un utilisateur

Énoncé : Enfin, nous ajoutons la fonctionnalité de suppression d'un utilisateur.

1/Ajouter le modal de suppression

```
<!-- Modal Confirmer Suppression -->
<div class="modal fade" id="deleteUserModal" tabindex="-1" aria-labelledby="deleteUserModalLabel" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="deleteUserModalLabel">Supprimer un utilisateur</h5>
```

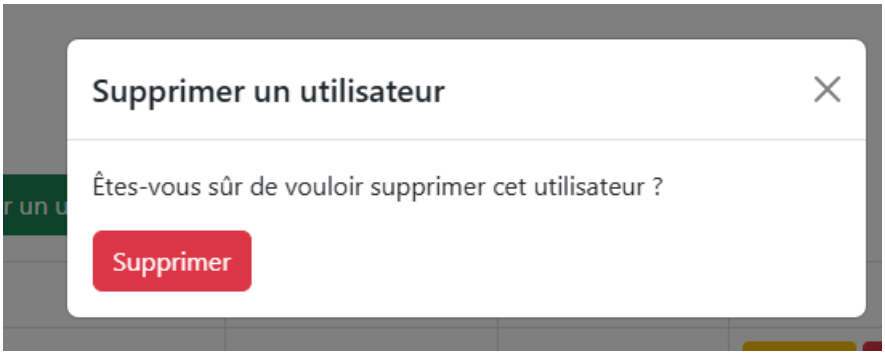
```
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
      </div>
      <div class="modal-body">
        <p>Êtes-vous sûr de vouloir supprimer cet utilisateur ?</p>
        <button id="confirmDeleteUser" class="btn btn-danger">Supprimer</button>
      </div>
    </div>
  </div>
</div>
```

2/Code de l'action de la supprission :

```
const deleteModal = new bootstrap.Modal(document.getElementById("deleteUserModal"));


function showDeleteModal(id) {
      deleteUserId = id;
      deleteModal.show();
    }

confirmDeleteUser.onclick = async function () {
  try {
    await fetch(`${apiBaseUrl}/${deleteUserId}`, {
      method: "DELETE"
    });
    deleteModal.hide();
    fetchUsers();
  } catch (error) {
    console.error("Erreur lors de la suppression d'un utilisateur :", error);
  }
};
```

Explication :

- Lorsqu'un utilisateur souhaite supprimer un utilisateur, nous envoyons une requête DELETE à l'API pour supprimer cet utilisateur.
- La liste des utilisateurs est actualisée après la suppression.

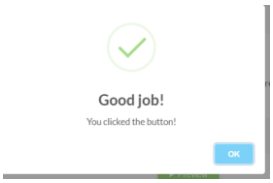


7.Actualiser la liste des utilisateurs dés que le chargement de la page

1/Ajouter ce code á la fin

```
fetchUsers(); // Charger les utilisateurs au premier chargement
```

2/utiliser la bibliothèque https://sweetalert.js.org/ pour gérer les notifications



Conclusion :

Cet atelier vous permet de créer une application simple de gestion des utilisateurs avec HTML, Bootstrap, et JavaScript. Vous avez appris à intégrer des modaux, à interagir avec une API, et à gérer les actions de CRUD (Créer, Lire, Mettre à jour, Supprimer).

Code final

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Gestion des Utilisateurs</title>
  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Bootstrap JS -->
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"></script>
  <script src="https://unpkg.com/sweetalert/dist/sweetalert.min.js"></script>
```

```
</head>

<body>
  <div class="container mt-5">
    <h1 class="text-center mb-4">Gestion des Utilisateurs</h1>
    <!-- Boutons pour ajouter et actualiser -->
    <button class="btn btn-success mb-3" data-bs-toggle="modal" data-bs-target="#addUserModal">Ajouter
un
      utilisateur</button>
    <button class="btn btn-primary mb-3" onclick="fetchUsers()">Actualiser la liste</button>
    <!-- Table des utilisateurs -->
    <table class="table table-bordered">
      <thead>
        <tr>
          <th>ID</th>
          <th>Nom</th>
          <th>Email</th>
          <th>Actions</th>
        </tr>
      </thead>
      <tbody id="userTableBody">
        <!-- Les utilisateurs seront chargés ici dynamiquement -->
      </tbody>
    </table>
  </div>

  <!-- Modal Ajouter Utilisateur -->
  <div class="modal fade" id="addUserModal" tabindex="-1" aria-labelledby="addUserModalLabel" aria-
hidden="true">
    <div class="modal-dialog">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title" id="addUserModalLabel">Ajouter un utilisateur</h5>
          <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
        </div>
        <div class="modal-body">
          <form id="addUserForm">
            <div class="mb-3">
              <label for="addUserName" class="form-label">Nom</label>
              <input type="text" class="form-control" id="addUserName" required>
            </div>
            <div class="mb-3">
              <label for="addUserEmail" class="form-label">Email</label>
              <input type="email" class="form-control" id="addUserEmail" required>
            </div>
            <button type="submit" class="btn btn-primary">Ajouter</button>
          </form>
        </div>
      </div>
    </div>
  </div>

  <!-- Modal Modifier Utilisateur -->
  <div class="modal fade" id="editUserModal" tabindex="-1" aria-labelledby="editUserModalLabel" aria-
hidden="true">
    <div class="modal-dialog">
      <div class="modal-content">
```

```
        <div class="modal-header">
          <h5 class="modal-title" id="editUserModalLabel">Modifier un utilisateur</h5>
          <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
        </div>
        <div class="modal-body">
          <form id="editUserForm">
            <input type="hidden" id="editUserId">
            <div class="mb-3">
              <label for="editUserName" class="form-label">Nom</label>
              <input type="text" class="form-control" id="editUserName" required>
            </div>
            <div class="mb-3">
              <label for="editUserEmail" class="form-label">Email</label>
              <input type="email" class="form-control" id="editUserEmail" required>
            </div>
            <button type="submit" class="btn btn-primary">Modifier</button>
          </form>
        </div>
      </div>
    </div>
  </div>

  <!-- Modal Confirmer Suppression -->
  <div class="modal fade" id="deleteUserModal" tabindex="-1" aria-labelledby="deleteUserModalLabel"
    aria-hidden="true">
    <div class="modal-dialog">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title" id="deleteUserModalLabel">Supprimer un utilisateur</h5>
          <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
        </div>
        <div class="modal-body">
          <p>Êtes-vous sûr de vouloir supprimer cet utilisateur ?</p>
          <button id="confirmDeleteUser" class="btn btn-danger">Supprimer</button>
        </div>
      </div>
    </div>
  </div>

  <script>
    const apiBaseUrl = "https://ca2afeac728b05c101d7.free.beeceptor.com/api/utilisateurs/";
    const userTableBody = document.getElementById("userTableBody");

    function escapeString(str) {
      return str.replace(/'/g, "\\'").replace(/"/g, '\\"');
    }

    async function fetchUsers() {
      try {
        const response = await fetch(apiBaseUrl);
        const users = await response.json();
        renderUsers(users);
      } catch (error) {
        console.error("Erreur lors du chargement des utilisateurs :", error);
      }
    }
```

```
    function renderUsers(users) {
      userTableBody.innerHTML = users.map(user => `
    <tr>
      <td>${user.id}</td>
      <td>${user.name}</td>
      <td>${user.email}</td>
      <td>
        <button class="btn btn-warning btn-sm" onclick="showEditModal('${user.id}',
'${escapeString(user.name)}', '${escapeString(user.email)}')">Modifier</button>
        <button class="btn btn-danger btn-sm" onclick="showDeleteModal('${user.id}')">Supprimer</button>
      </td>
    </tr>
  `).join('');
    }

    const addUserForm = document.getElementById("addUserForm");
    const addModal = new bootstrap.Modal(document.getElementById("addUserModal"));

    addUserForm.onsubmit = async function (e) {
      e.preventDefault();
      const name = document.getElementById("addUserName").value;
      const email = document.getElementById("addUserEmail").value;

      try {
        await fetch(apiBaseUrl, {
          method: "POST",
          headers: { "Content-Type": "application/json" },
          body: JSON.stringify({ name, email })
        });
        addModal.hide();
        fetchUsers();
        swal("Good job!", "Utilisateur ajouté avec succés", "success");
      } catch (error) {
        console.error("Erreur lors de l'ajout d'un utilisateur :", error);
      }
    };

    const editModal = new bootstrap.Modal(document.getElementById("editUserModal"));

    function showEditModal(id, name, email) {
      document.getElementById("editUserId").value = id;
      document.getElementById("editUserName").value = name;
      document.getElementById("editUserEmail").value = email;
      editModal.show();
    }

    editUserForm.onsubmit = async function (e) {
      e.preventDefault();
      const id = document.getElementById("editUserId").value;
      const name = document.getElementById("editUserName").value;
      const email = document.getElementById("editUserEmail").value;

      try {
        await fetch(` ${apiBaseUrl}/${id}` , {
          method: "PUT",
          headers: { "Content-Type": "application/json" },
          body: JSON.stringify({ name, email })
```

```
        });
        editModal.hide();
        fetchUsers();
      } catch (error) {
        console.error("Erreur lors de la modification d'un utilisateur :", error);
      }
    };

    const deleteModal = new bootstrap.Modal(document.getElementById("deleteUserModal"));

    function showDeleteModal(id) {
      deleteUserId = id;
      deleteModal.show();
    }

    confirmDeleteUser.onclick = async function () {
      try {
        await fetch(` ${apiBaseUrl}/${deleteUserId}` , {
          method: "DELETE"
        });
        deleteModal.hide();
        fetchUsers();
      } catch (error) {
        console.error("Erreur lors de la suppression d'un utilisateur :", error);
      }
    };

    fetchUsers(); // Charger les utilisateurs au premier chargement
  </script>

</body>

</html>
```